ACADEMY PORTAL

V1.0

Anusha Kula Sekar

Vani K

Java Mode 2 Training – Final Hackathon

# Contents

Table of Contents

[Contents 2](#_Toc86671974)

[UML diagram 4](#_Toc86671975)

[Database Design 5](#_Toc86671976)

[User 5](#_Toc86671977)

[Schema 5](#_Toc86671978)

[Sample Data 5](#_Toc86671979)

[FACULTY 6](#_Toc86671980)

[UI SCREENS 8](#_Toc86671981)

[Register as faculty 9](#_Toc86671982)

[Login 10](#_Toc86671983)

[Login – Initial 10](#_Toc86671984)

[Login – Error 1 11](#_Toc86671985)

[Login – Error 2 12](#_Toc86671986)

[Faculty List 14](#_Toc86671987)

[REST API design: 15](#_Toc86671988)

[Entity objects (Models – tables in database) 15](#_Toc86671989)

[Controllers 15](#_Toc86671990)

[Respository 16](#_Toc86671991)

[Faculty 17](#_Toc86671992)

[Controllers 17](#_Toc86671993)

[Respository 19](#_Toc86671994)

[Sequence Diagram 23](#_Toc86671995)

User Stories

Academy\_Portal \_User\_story\_01: Login: User have to enter username and password and click on login button and then user will be taken to homepage based on their role.Error Messages: "Invalid username or password.Please enter correct username and password".NFR: Performance -Response time should be less than a second. SonarQube - Maintainability, Reliability, Security should be in green.

Academy\_Portal \_User\_story\_02: Register as Faculty:on clicking register hyperlink ,User will be redirected to faculty registration page and then faculty have to provide Name,Qualification,Location,DOJ,password.All fields are mandatory.After entering valid values and click on "Submit" button ,it show the "Faculty details added successfully" msg. Error Messages: "You have entered invalid details...Please enter correct details." or against each field we can show the message. NFR: Performance -Response time should be less than a second. SonarQube - Maintainability, Reliability, Security should be in green

Academy\_Portal \_User\_story\_03: Register as a Student: User will be redirected to register as a student page and then student have to provide First Name, Last Name, Age, Gender, Contact Number, Email id,password.All fields are mandatory.After entering valid values and click of "Submit" button show the "Student details added successfully" msg. Error Messages: "You have entered invalid details...Please enter correct details." or against each field we can show the message. NFR: Performance -Response time should be less than a second. SonarQube - Maintainability, Reliability, Security should be in green.

Academy\_Portal \_User\_story\_04: Faculty pending approval by admin/faculty: Admin/Faculty will be redirected to list of faculties to be approved .On Clicking approve button ,the status of faculty will be changed to active and alert message will be shown as “faculty approved”. NFR: Performance -Response time should be less than a second. SonarQube - Maintainability, Reliability, Security should be in green.

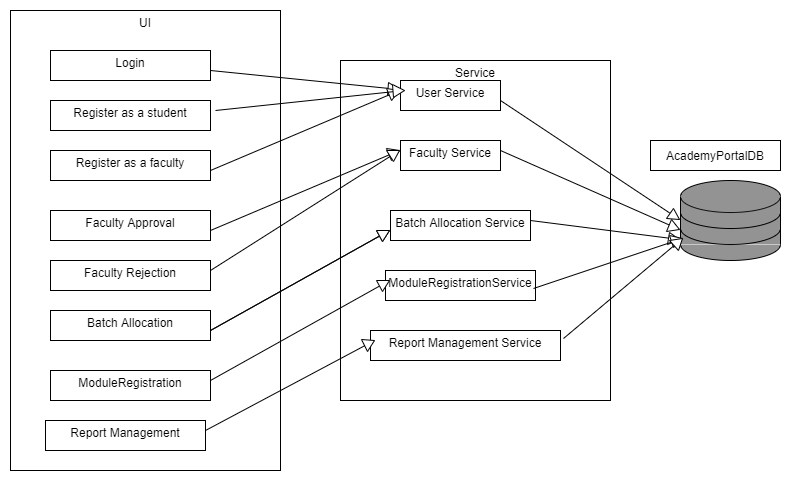
Academy\_Portal \_User\_story\_05: Faculty pending rejected by admin/faculty: Admin/Faculty will be redirected to list of faculties to be approved .On Clicking reject button ,the status of faculty will remains same and alert message will be shown as “faculty Rejected”. NFR: Performance -Response time should be less than a second. SonarQube - Maintainability, Reliability, Security should be in green.

Academy\_Portal \_User\_story\_06:Module Registration:Admin on logging into the system selects module registration option.On click module registration button ,module registration screen will be open.Admin has to enter course name,faculty id,duration,course type.All fields are mandatory,after entering valid details and click on “Submit ” button,It shows “Module registered successfully.Error message:” You have entered invalid details,please enter correct details”.or against easc field we can show the message. NFR: Performance -Response time should be less than a second. SonarQube - Maintainability, Reliability, Security should be in green.

Academy\_Portal \_User\_story\_07:Batch Allocation:Admin on logging have option to batch allocation .On clicking batch allocation hyperlink,Admin should enter the batch name,duration,maximum\_students,facultyname. All fields are mandatory,after entering valid details and click on “Submit ” button,It shows “Batch registered successfully”.Error message:” You have entered invalid details,please enter correct details”.or against each field we can show the message. NFR: Performance -Response time should be less than a second. SonarQube - Maintainability, Reliability, Security should be in green.

Academy\_Portal \_User\_story\_08: Report Management: Admin on logging have option for Report Management.On clicking report hyperlink,Admin should enter the feedback details.All fields are mandatory,after entering valid details and click on “Submit ” button, credit will be updated to the faculty based on the feedback. Error message:” You have entered invalid details,please enter correct details”.or against each field we can show the message. NFR: Performance -Response time should be less than a second. SonarQube - Maintainability, Reliability, Security should be in green.

# UML diagram



.

# Database Design

## User

### Schema

|  |  |  |
| --- | --- | --- |
| USERID | Integer | Primary key , Auto Increment |
| FIRST NAME | Varchar(15) | Not null , Unique |
| LAST NAME | Varchar(20) | Not null |
| ROLES | Varchar(30) | Not null |
| AGE | Integer | Not null |
| GENDER | Varchar(20) | Not null |
| EMAIL ID | Varchar(20) | Not null, Unique |
| PASSWORD | Varchar(20) | Not null |
| CREDIT | Integer | Not null |
| STATUS | Varchar(20) | Not null |
| BATCH ID | Integer |  |

### Sample Data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| USERID | USERNAME | PASSWORD | ROLES | STUDENT\_ID | FACULTY\_ID |
| 1 | Praveena | P@199FDg | Faculty | Null | 1718150 |

## BATCH

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| BATCH\_ID | Int | Primary key, Autoincreament |
| BATCHNAME | Varchar(10) | Not null , Unique |
| FROMDATE | DATE | Not null |
| TODATE | DATE | Not null |
| MAXIMUM\_STUDENTS | Varchar(12) | Not null |
| FACULTY\_ID | Varchar(30) |  |

Sample Data

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Faculty\_Id | First Name | Last Name | Qualification | Date\_of\_  joining | Location | Password | Credit | Status | Batch\_ID |
| 5167101 | Thirisha | R | Ph.D | 2021-02-10 | chennai | thr@sv$h | 0 | inactive | 101 |

MODULE REGISTRATION

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| MODULE\_ID | Int | Primary key, Autoincreament |
| MODULENAME | Varchar(10) | Not null , Unique |
| FROMDATE | Date | Not null |
| TODATE | Date |  |
| FACULTY\_ID | Varchar(20) |  |
| MODULE\_TYPE | Varchar(20) | Not null |

Sample Data

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| STUDENT\_ID | FIRST  NAME | LAST  NAME | PASSWORD | AGE | GENDER | CONTACT  NUMBER | EMAIL |
| 1718151 | Vani | K | VANI@KV200 | 21 | FEMALE | 9597271226 | [VANI.KV200@GMAIL.COM](mailto:VANI.KV200@GMAIL.COM) |

# REPORT

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **REPORTID** | **INT** | **Primary key, Autoincreament** |
| **INTERACTION** | **INT** | **Not null , Unique** |
| **PARTICIPATION** | **INT** | **Not null** |
| **UNDERSTANDING** | **INT** | **Not null** |
| **IMPLEMENTATIOM** | **INT** | **Not null** |
| **bATCHID** | **INT** |  |

# 

# Sample Data

# 

# 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| REPORTID | INTERACTION | PARTICIPATION | UNDERSTANDING | IMPLEMENTATION | BATCHID |
| 101 | 5 | 5 | 5 | 5 | 1 |

# 

# ER Diagram

1:1

Modules

Id:int

modulename:varchar

startDate:date

endDate:date

moduleType:varchar

facultyId:int

1:1

Report

Id: int

Participation:int

Understanding:int

Implementation:int

Interaction:int

BatchID:int

1:1

User(faculty/student/admin)

Id: int

firstName:varchar

lastname:varchar

role:varchar

age:int

gender: varchar

email: varchar

credit:int

status: varchar

password: varchar

batchId:int

Batch

BatchId:int

BatchName:varchar

startDate:date

endDate:date

MaximumStudents:int

FacultyName:varchar

FacultyId:int

# UI SCREENS

**Server URL**: GET:<http://localhost:4200/>home

**Server Response**:– http status -ok

|  |  |  |
| --- | --- | --- |
| Welcome to Academy Portal  Home About us Sign in | | |
|  | | |
|  | New User? Register as faculty/ register as student |  |
| Java Mode 2 Training – Final Case Study | | |



# Register as faculty

**Server URL**: POST <http://localhost:7070/user>

**Server Request**:

{ “firstname”:”anu”,”lastname”:”s”,”password”:”yyiutgfvs”,”role”:’faculty”,”email”:”anu@gmail.com”}

**Server Response**: {“msg”:“success”} – http status -ok

|  |
| --- |
| Welcome to Academy portal |
| Register  FirstName lastname password  Role  Email Id  Register Reset |
| Java Mode 2 Training – Final Case Study |



## Login

### Login – Initial

**Server URL**: POST <http://localhost:4200/login>

**Server Request**:

{

  "username": "admin",

  "password": "djfjdi9we"

}

**Server Response**: {“msg”:“success”} – http status -ok

|  |
| --- |
| Welcome to Academy Portal |
| Login  <Username>  <password>  Login Reset New  New user?Register here |
| Java Mode 2 Training – Final Case Study |

### Login – Error 1



**Server URL**: POST <http://localhost:4200/login>

**Server Request**:

{ “username”:”vani”,”password”:}

**Server Response**: http status-Not ok

|  |
| --- |
| Welcome to Academy Portal |
| Login  <Username> Username is required  <password> Password is required  Login Reset New User? SignUp |
| Java Mode 2 Training – Final Case Study |



### Login – Error 2

Server URL: http://localhost:4200/login

Server Request: {”username”:”sample111111”, “password" :"jd3lk3rn3"} Server Response: {“msg”:“ “User Account does not exits”} – http status -not ok

|  |
| --- |
| Welcome to Academy Portal |
| User Account does not exists  Login  <Username>  <password>  Login Reset New User? SignUp |
| Java Mode 2 Training – Final Case Study |



Faculty List to be Approved

Server URL: GET:<http://localhost:7070/faculty?role='+role+'&status='+status>

Server Request: {’role”:”faculty”,status":”inactive”}

Server Response: http-status-ok



Java Mode 2 Training – Final Case Study

Welcome to Academy Portal

List of Faculty- For Approval

Logout

Module Registration

Batch List

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Name | Qualification | Location | Date\_of\_Joining | Credit | Batch\_Id | Action |
| 1 | vani | MBA | Chennai | 2020-01-03 | 4 | 101 | Accept Reject |
| 2 | Anu | Ph.D | Salem | 2020-02-01 | 0 | 102 | Accept Reject |

## Faculty List

Server URL: GET:<http://localhost:7070/user?id='+id>

Server Request: {”id":”1”}

Server Response: http-status-ok



|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Welcome to Academy Portal System | | | | | | | | | |
|  | Welcome Faculty | | | Logout | | | | | |
|  |  | | | | | | | |  |
|  | | Id | Name | | Qualification | DOJ | Location | Status |  |
| 1 | Vani | | MBA | 2021-01-  02 | Chennai | Active |
|  | | | | | | | | | |
| Java Mode 2 Training – Final Case Study | | | | | | | | | |

## REST API design: User Service

### Entity objects (Models – tables in database)

**public class** User {

@Id

@GeneratedValue(strategy = GenerationType.***AUTO***)

**private int id;**

**private String firstName;**

**private String lastName;**

**private String roles;**

**private int age;**

**private String gender;**

**private String emailId;**

**private String password;**

**private int credit;**

**private String Status;**

**private int BatchId;**

}

### Controllers

**public class** UserController {

@Autowired UserService userserv;

@GetMapping("/users")

**public** ResponseEntity<List> users() {

**return new** ResponseEntity<List>(userserv.findAll(), HttpStatus.***OK***);

}

@GetMapping("/user")

**public** ResponseEntity<User> user(@RequestParam **int** id) {

**return new** ResponseEntity<User>(userserv.findById(id), HttpStatus.***OK***);

}

@PostMapping("/user")

**public** ResponseEntity<User> createuser(@RequestBody User user) {

**return new** ResponseEntity<User>(userserv.save(user), HttpStatus.***OK***);

}

@PutMapping("/user")

**public** ResponseEntity<User> updateuser(@RequestBody User user) {

**return new** ResponseEntity<User>(userserv.save(user), HttpStatus.***OK***);

}

@DeleteMapping("/user")

**public** ResponseEntity<String> deleteuser(@RequestParam **int** id) { userserv.deleteById(id);

**return new** ResponseEntity<String>("Success", HttpStatus.***OK***);

}

}

#### Services

**public interface** UserService { **public** User findById(**int** id); **public** List<User> findAll(); **public** User save(User user);

**public boolean** deleteById(**int** id);

}

@Service

**public class** UserServiceImpl **implements** UserService {

@Autowired UserRepository userrepo;

@Override

**public** User findById(**int** id) {

**return** userrepo.getById(id) ;

}

@Override

**public** List<User> findAll() {

**return** userrepo.findAll();

}

@Override

**public** User save(User user) {

**return** userrepo.save(user);

}

@Override

**public boolean** deleteById(**int** id) { userrepo.deleteById(id); **return true**;

}

}

### Respository

public interface UserRepository extends JpaRepository<User, Integer> {

}

Module Registration Service

public class ModuleRegistration {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private int id;

private String moduleName;

private int duration;

private String moduleType;

private int facultyId;

}

Controllers

public class ModuleRegController {

@PostMapping(“/module”)

public ResponseEntity<Module> module(){

// call service method

//ResponseEntity<>( service.save(module),HttpStatus.OK);

//ResponseEntity<>(HttpStatus.NOT\_FOUND)

}

public class ModuleFacultyController {

@GetMapping("/facultyId")

public ResponseEntity<List> facultyId(@RequestParam String facultyId) throws URISyntaxException {

return new ResponseEntity<List>(facultyService.findAllByfacultyIds(facultyId), HttpStatus.OK);

}

}

#### Services

**public interface** ModuleService { **public** User findById(**int** id); **public** List<User> findAll(); **public** User save(User user);

**public boolean** deleteById(**int** id);

}

### Respository

public interface ModuleRepository extends JpaRepository<Module, Integer> {

}

Class Diagram

User

Diagram

Description automatically generated

MODULE

Diagram

Description automatically generated

BATCH

Diagram

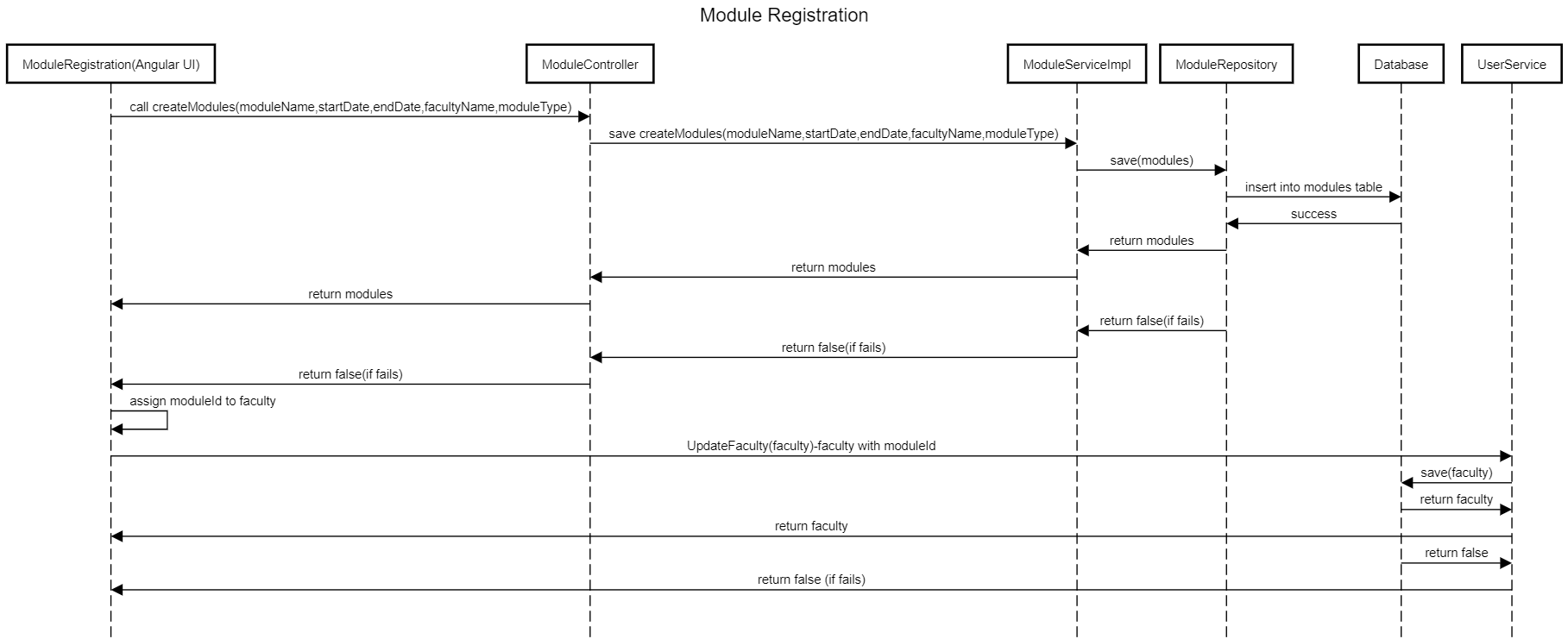
Description automatically generated

# REPORT

# Diagram Description automatically generated

# Sequence Diagram

1.Module Registration by Admin



2.Batch Allocation by Admin

Diagram

Description automatically generated

Task Allocation

|  |  |
| --- | --- |
| UI/Service | Developers |
| UI | Vani, Anusha |
| UserService | Vani |
| BatchAllocationService | Anusha |
| ModuleRegistrationService | Anusha |
| ReportManagementService | Vani |